〈研究論文〉

Journal of the Korean Institute of Industrial Engineers
Vol. 14, No. 1, June 1988

自動生産体制(FMS)에서의 生産日程計劃

(A Scheduling Heuristic Alogorithm for Flexible Manufacturing Systems)

魯 仁 珪*
崔 政 祥*

Abstracts

This research is concerned with production scheduling for FMS (Flexible Manufacturing System) which consists of machine centers served by cycle conveyor. The objective of the ressarch is to develop and evaluate scheduling procedures to minimze the mean flow time. An optimal algorithm called SCTF (Shortest Cycle Time First) is proposed when the conveyor runs at minimum possible speed (CS=1) and a heuristic algorithm called SCTJMF (Shortest Cycle Time and Job Matching Algorithm) is suggested when the conveyor runs at double speed (CS=2).

The evaluation of the heuristic algorithm was implemented by comparision with the optimal algorithm for 112 experimentations for CS=1 and random schedule. The results showed that the proposed heuristic algorithm provides better solution that can be regarded noticeable when compared with SCTF algorithm and random scheduling.

1. 서 론

소비자의 제품에 대한 요구가 갈수록 다양해져서 같은 용도의 재품이라도 다양한 모델이 필요하게 되었고 이에 따라 제품의 수명기간(Life-Cycle)이 점점 더 짧아지고 있다.

따라서 종래의 흐름생산(Flow-Shop)이나 로트생산과 같이 생산성만을 추구하는 양산시스템에서는 제품의 수요가 급격히 변할 경우에는 대형설비와 다량의 재고품을 안게되는 등 제품수요 및 소비자 요구에 대처하는 것이 늦어 많은 낭비와 손실이 생기게 된다. 제조공업에서는 이러한 수요의 다양한 변화에 신속하게 대처할 수 있도록 생산성과 설비의 효율을 높이는 동시에, 제품에 대한 신뢰성을 확보함으로써 필요한 제품을 적시적정생산할 수 있는 자동생산체제

* 한양대학교 공과대학 산업공학과

(Flexible Manufacturing System : FMS)가 절실히 필요하게 되었다.

따라서 본 연구의 목적은 자동생산체제에서 평균처리시간을 최소화하기 위한 발견적 기법을 개발하고 개발한 발견적 기법의 유효성을 평가하는데 있다.

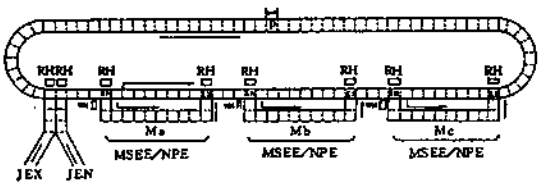
FMS는 1968년에 영국에서 처음으로 연구개발이 시작되어 80년대에 들어와서 부터 그에 대한 연구와 개발이 매우 활발하게 이루어지고 있다[8], [11].

FMS는 생산일정 계획에 대한 연구로는 Nof, Barash, Solberg[10]과 Stecke[16]의 최적화 이론을 이용한 연구, 대기네트워크 이론을 이용한 연구 [12], 시뮬레이션을 이용한 Stecke과 Solberg의 연구 [15]와 발견적 기법을 이용한 대표적 연구로는 Galouzeau, M. and Siegel, P.의 연구 [4], Kusiak의 연구 [9], C. Sriskandarajah, P. Ladet and R. Germain의 연구 [14]를 비롯한 많은 연구들이 있다.

본 연구에서는 순환형 Conveyor가 장착된 자동생산체제(FMS)에서 Conveyor Speed가 Low-Speed일 때 평균처리시간을 수행지수로 선정하고, 수행지수인 평균처리시간을 최소화하는 발견적 기법을 개발하였다.

2. 모델 설정

본 연구의 모델로 이용한 Manufacturing System은 [그림 1]과 같이 순환형 Conveyor로 장착된 M대의 기계에서 n개의 job를 가공해야 하는 생산라인의 경우로써 Conveyor의 방향이 단일방향으로 정해진 Job-Shop의 일종이다.



[그림1] A manufacturing system with three MSEE/NPS type machine centers

[그림1]에 있는 부호의 의미는 다음과 같다.

MSEE/NPE : Machine center with Sperate Entrance and Exit, where jobs leave from the machine center with Nonpriority Exit.
JEN : Job Entrance to the system
JEN : Job Exit from the ststem
EN : Machine center Entrance
EX : Machine center Exit
RH : Read Head which reads job types
WH : Write Head which wtites remaining jobs after treatment
P : Pallet

앞에서 제시한 본 연구의 모델에서 Machine Center가 3개일때 가능한 job type의 종류는 다음과 같은 15가지가 있으며, 여기서 job type의 수는 각 Machine Center에서 처리되는 task들의 조합의 수이다.

Jcba, Jcab, Jbca, Jbac, Jacb, Jcb,
Jca, Jba, Jab, Jac, Ja, Jb, Jc, Jabc

각 job type에서 앞에 있는 task가 뒤에 있는 task에 우선하는 precedence rule이 적용된다. 즉, machine center가 [그림1]처럼 a, b, c 순으로 설치되어 있는 Manufacturing System에서 Jcba를 처리할 때 machine a가 맨 처음에 놓여 있을지라도 먼저 machine c에서 task Jc를 처리하고 나서 machine b에서 task Jb를 처리할 수 있으며 task Jc와 Jb를 처리하고 나서 machine a에서 task Ja를 처리 하게 된다.

따라서 job의 형태에 따라 똑같은 task의 수를 갖는 job이라 할지라도 소요되는 시간(Cycle Time : CT)이 다르게 된다. 각 job type을 처리하는데 소요되는 시간을 살펴 보면 다음과 같다.

CT=3 : Jcba=Jc < Jb < Ja

CT=2 : Jcab=Jc < Jba
Jbca=Jbc < Ja
Jbac=Jb < Jac
Jbcb=Jac < Jb
Jcb=Jc < Jb
Jca=Jc < Ja
Jba=Jb < Ja

CT=1 : Jab, Jac, Jbc, Ja, Jb, Jc, Jabc

여기서 부등호 < 는 「선 < 후」를 의미한다.

2.1 전제조건

모델을 설정하기 위해 다음과 같은 전제조건을 제시한다.

(1) 모든 작업은 시간 0에서 시작이 가능하다.
(2) 각 기계는 여러 작업을 동시에 가공할 수 없다.
(3) 가공중인 작업은 그 기계에서 가공이 끝날 때까지 제거되지 않는다. (Non-preemption)
(4) 기계와 Conveyor는 공정중에 고장은 없다.
(5) GT(Group Technology)개념에 의해 processing-time이 같은 task들도 그룹화 되어 있다.
(6) 각 job은 한개 이상의 task로 이루어져 있다.
(7) 각 작업들간에 Priority는 없다.
(8) 각 job안에서 앞에 있는 task가 뒤에 있는 task에 우선한다. (Precedence rule)
(9) job이 기계에서 처리를 받고 다시 unloading될 때는 loading되기 전에 실렸던 팔레트에 unloading 되어야 한다.

2.2 부호설명

M : machine center의 수
a,b,c : machine center의 고유인덱스
i : job type의 인덱스
N : 처리해야 할 전체의 job수
TT : 한 task가 기계에 loading 되어서 처리를 받고 다시 pallet에 unloading될 때까지 소요되는 시간 (TT=loading time+processing time +unloading time)
CS : Conveyor speed(CS=Pallet/TT) (CS 1/TT)
FT : CS=2인 경우 Job Matching이 되는 두 job type중 먼저 배열해야 하는 job type
n(FT) : FT에 있는 job의 갯수
SD : CS=2인 경우 Job Matching이 되는 두 job type중 뒤에 배열해야 하는 job type
n(SD) : SD에 있는 job의 갯수
n(R) : 맨 처음에 주어진 job중 배열하고 현재 남아 있는 job의 총
CT : Conveyor가 한바퀴 도는데 걸리는 시간
ℓ : lost job
CS=2(혹은 CS=3)일때 이웃하는 Pallet에 같은 기계에서 처리해야 하는 task를 갖는 job을 배열할 경우 두 job중 한 job은 그 기계에서 처리받지 못하고 Pallet에 실린 채로 한 Cycle을 돌게 되는 job을 말한다.
Fij : job type i의 j번째 job을 처리하는데 소요되는 시간
Fi : job type i를 모두 처리하는데 소요되는 총처리시간 ($\sum_j$ Fij)
F : 전체의 job를 모두 처리하는데 소요되는 총처리시간 ($\sum_i \sum_j$ Fij)
$\overline{F}$: 전체의 job를 모두 처리하는데 소요되는 평균처리시간

2.3 수리적모델의 설정

job type i를 처리 하는데 소요되는 총처리시간 Fi는 식(1)과 같이 쓸 수 있다.

$$Fi = \Sigma Fij \quad \cdots\cdots (1)$$

주어진 job 전체를 처리하는데 소요되는 총처리시간 F는 다음 식(2)와 같다.

$$F = \Sigma F1j + \Sigma F2j + F3j + \cdots\cdots + \Sigma F15j \quad (2)$$

주어진 job 전체를 처리하는데 소요되는 평균처리시간 F는 식 (3)과 같이 나타낼 수 있다.

$$\overline{F} = F/(N^{*}CS) \quad \cdots\cdots (3)$$

본 연구의 목적은 모든 작업을 처리하는데 소요되는 평균처리시간을 최소화하는 것이므로 앞에서 제시한 부호 및 수식들을 이용하여 다음 식 (4)와 같은 목적함수를 도출할 수 있다.

$$\text{Minimize } \overline{F} = F/(N^{*}CS) \quad \cdots\cdots (4)$$

3. 알고리듬의 개발

3.1 CS=1 일때의 알고리듬의 개발

Conveyor Speed(CS) CS=Pallet/TT이므로 CS는 한 task의 처리시간 TT에 반비례한다.

왜냐하면 job이 Conveyor상의 pallet에 실려서 어느 기계에서 처리를 받고 unloading될 때도 반드시 loading되기 전에 놓였던 pallet에 다시 옮겨져야 하기 때문이다.

CS=1이라는 말은 한 task를 한 기계에 loading해서 처리하고 unloading 하는데 소요되는 시간 TT=lt+pt+ut=1(unit time)인 경우를 말하는 것으로서 Conveyor Speed가 가장 낮은 경우이다. CS=1일때 평균처리시간을 최소화하는 Algorithm으로 다음과 같은 SCTF(Shorest Cycle Time First) Algorithm을 제시한다.

3.1.1 SCTF Algorithm

SCTF Alogorithm은 기존의 SPT Rule와 동일한 방법으로 구체적인 내용은 다음 Step에서 설명하는 바와 같으며 CS=1일 때 평균처리시간을 최소화하고자 할때 SCTF Algorithm을 적용하면 optimal 값을 얻을 수 있다. 그에 대한 증명은 기존의 SPT Rule과 같은 경우이므로 생략한다[1].

[단계 1] 주어진 job type중 CT이 가장 작은것 즉 CT=1인 것이 있으면 CT=1인 job을 먼저 배열한다.
CT=1인 job이 없으면 [단계 2]로 가라.

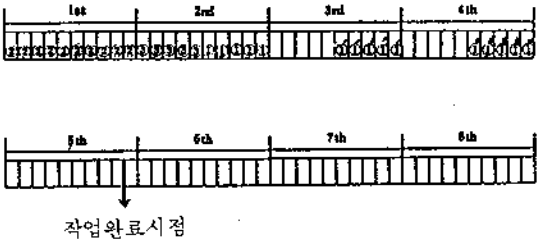
[단계 2] 주어진 job type중 CT=2인 것이 있으면 CT=2인 job을 배열하고 만약 없으면 [단계 3]으로 가라.

[단계 3] 주어진 job type중 CT=3인 것이 있으면 CT=3인 job을 배열하고 만약 없으면 배열을 끝내고 [단계 4]로 가라.

[단계 4] 평균 처리시간을 구한다.

본 연구에서 모델로 이용한 자동생산체제(FMS)의 구성 여건이 다음과 같을때의 SCTF Algorithm을 적용하여 예제를 풀어 보면 [단계 2]과 같다.

1) M=3　　2) CS=1　　3) P=10개
4) ①Jcba : 5개,　②Ja : 10개,　③Jb : 5개



[그림2] A schedule with SCTF algorithm applied for CS=1

여기서 ①′는 주어진 job type이 CT=2나 CT=3인 job일때 첫번째 Cycle에서 처리 받고난 후에 남아 있는 job의 형태이고, ①″는 주어진 job type이 CT=3인 job일때 첫번째 Cycle과 두번

째 Cycle에서 처리 받고 난 후에 남아 있는 job의 형태를 의미한다. 이 때의 평균처리시간을 구하면 다음과 같다.

F1=45+46+47+48+49=235

F2=10+11+12+13+14+15+16+17+18+19=145

F3=20+21+22+23+24=110

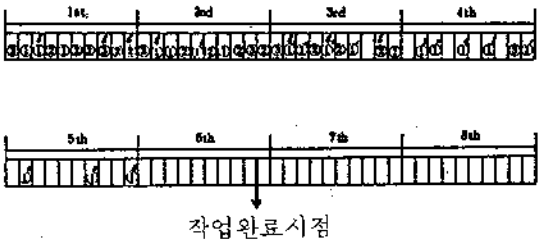
N=n1+n2+n3=20

F=F1+F2+F3=490

$\overline{F}$=F/(N*CS)=490/(20*1)=24.5

3.2 CS=2일 때의 발견적 기법 개발

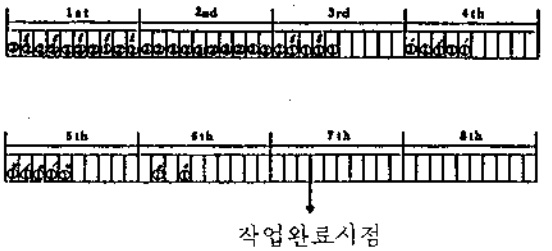
SCTF Algorithm이 CS=1일 때는 optimal 임을 알 수 있었다. [그림2]의 예제와 동일한 여건으로 CS=2일 때 SCTF Algorithm을 적용하여 Schedule해서 Random Schedule과 비교해 보았더니 다음과 같았다.



[그림3] A Random Schedule for CS=2

[그림2]에서와 같은 방법으로 [그림3]의 Random Schedule에서 평균처리시간을 구하면

$\overline{F}$=F/(F*CS)=668/(20*2)=16.7



[그림4] A SCTF Algorithm Schedule for CS=2

위 [그림4]의 SCTF Algorithm Schedule에서 평균처리시간을 구하면

$\overline{F}$=F/(N*CS)=595/(20*2)=14.9

앞 [그림3]와 [그림4]에서 보는 바와 같이 CS=2일 때 SCTF Algorithm을 적용한 Schedule이 Random Schedule보다 평균처리시간을 10.93%나 더 줄일 수 있었다. 즉 SCTF Algorithm이 CS=2일 때도 Random Schedule에 비해서 훨씬 우수하다.

그러나, CS=2일 때는 SCTF Algorithm이 optimal이 아니다. CS=2일 때 SCTF Algorithm을 적용하여 Schedule하게 되면 Random Schedule 보다는 평균처리시간이 향상 되기는 하지만, [그림3]의 Random Schedle에서 보듯이 Conveyor상에서 lost job이 생기게 되고 그로 인해 평균 처리시간이 상당히 증가하게 된다.

왜냐하면 3.1에서 살펴본 바와 같이 CS=1일 때는 한 개의 task가 처리되는 동안 한 개의 Pallet가 이동하게 되므로 CS=1일 때는 어떤 형태의 job type이라 할지라도 Conveyor상에서 lost job의 발생없이 한 개의 job씩 정해진 순서대로 처리가 가능하며 이때 SCTF Algorithm을 적용하여 최적해를 구할 수 있었다.

그러나, CS=2로 증가하면 CS가 두배로 빨라지기 때문에 CS=1일 때 한 개의 task가 처리되는 동안 한 개의 Pallet가 이동하는 것과는 달리 한 개의 task가 처리되는 동안 두 개의 Pallet가 이동하게 된다.

그러므로 만약 CS=1일 때 처럼 SCTF Algorithm을 CS=2일 때도 적용하면 반드시 Conveyor 상에서 lost job이 생기게 되며 그로인해서 평균처리시간이 엄청나게 증가하게 되므로 CS=2일 때 SCTF Algorithm은 평균처리시간을 줄이려는 목적에 적절하지 못하다.

따라서 M=3이고 CS=2일 때 SCTF Algorithm을 적용했을때처럼 평균처리시간 $\overline{F}$에 심각

한 영향을 끼치는 Conveyor상에서의 lost job이 발생하는 것을 허용하지 않으면서 평균처리시간을 줄이는 기법으로 SCTJMF(Shortest Cycle Time and Job Matching First) Algorithm을 제시한다.

3.3 CS=2일 때 발견적 기법인 SCTJMF (Shortest Cycle Time and Job Matching First) Algorithm의 유도

SCTJMF Algorithm은 CS=2, M=3일 때 Conveyor 상에서 lost job을 허용치 않으면서 평균처리시간을 줄이려는 기법으로서 SCTF Algorithm에 Job Matching의 개념을 이용한 Algorithm이다.

Job Matching은 한 개의 task가 처리되는 동안 두 개의 Pallet가 이동하는 CS=2인 경우 서로 이웃하는 pallet에 같은 형태의 task가 놓이지 않도록 함으로써 lost job이 생기지 않도록 하기 위한 것이다.

예를 들어, Jcba와 Jbac의 경우를 보면, 다음 [표1]과 같다.

	CT	Machine Processed in the 1st CY	Machine Processed in the 2nd CY	Machine Processed in the 3rd CY
J cba	3	M c	M b	Ma
J bac	2	Mb	Ma, Mc	

〔표 1〕

An Example of Job Matching

위 [표1]에서 두 job의 처리 과정을 보면 각 Cycle에서 처리해야 할 두 job의 task가 동시에 한 기계에서 동시에 처리되어야 하는 경우가 없다. 이 처럼 두 job을 처리하는데 한 기계에서 처리해야 할 task가 중복되지 않게 짝지어졌을 때 완전한 Job Matching이 이루어졌다고 말할 수 있다. 같은 방법으로 M=3일 때 가능한 15가지 종류의 job type에 대해서 각 job들간의 Job Matching 관계를 살펴 보면 다음 [표2]와 같은 Job Matching Matrix(JMM)을 얻을 수 있다.

	Jabc	Jc	Jb	Ja	Jbc	Jac	Jab	Jba	Jca	Jcb	Jacb	Jbac	Jbca	Jcab	Jcba
Jabc	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Jc	0	0	1	1	0	0	1	1	0	0	0	1	0	0	0
Jb	0	1	0	1	0	1	0	0	1	1	1	0	0	1	1
Ja	0	1	1	0	1	0	0	1	1	1	0	1	1	1	1
Jbc	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
Jac	0	0	1	0	0	0	0	1	0	0	0	1	0	0	0
Jab	0	1	0	0	0	0	0	0	1	1	0	0	0	1	1
Jba	0	1	0	1	0	1	0	0	0	1	1	0	0	0	1
Jca	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0
Jcb	0	0	1	1	0	0	1	1	0	0	0	1	0	0	0
Jacb	0	0	1	0	0	0	0	1	0	0	0	1	0	0	0
Jbac	0	1	0	1	0	1	0	0	0	1	1	0	0	0	1
Jbca	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
Jcab	0	0	1	1	0	0	1	0	0	0	0	0	0	0	0
Jcba	0	0	1	1	0	0	1	1	0	0	0	1	0	0	0

〔표 2〕

Job Matching Matrix (JMM) for M = 3

3.4 SCTJMF(Shortest Cycle Time and Job Matching First) Algorithm

[단계 1] 처리를 필요로 하는 현재의 job type 중 CT이 가장 작은 것을 골라 FT(FT : First)에 둔다. 이때 만약 job type이 두개 이상일 때는 갯수가 제일 많은 것을 먼저 선택한다.

[단계 2] [표 2]에서 주어진 Job Matching Matrix를 이용하여 [단계 1]에서 선택된 job과 Job Matching이 이루어지는 job 중 CT가 제일 작은 것을 골라 SD(SD : Second)에 둔다. 이때 만약 job type이 두개 이상일 때는 갯수가 제일 많은 것을 먼저 선택한다.

[단계 3] [단계 1]과 [단계 2]에서 정해진 job을 배열하되 FT자리에 있는 job을 먼저 배열하고 SD 자리에 있는 job을 뒤에 번갈아 배열한다.

n(FT)=0, n(SD)=0, n(R)=0인지를 확인한다.

1) n(R)=0, n(FT)=0, n(SD)=0이면 [단계 4]로 가라.
2) n(R)=0, n(FT)=0, n(SD)=0이 아니면 [단계 5]으로 가라.
3) n(R)=0, n(FT)=0이 아니고, n(SD)=0이면 [단계 6]로 가라.
4) n(R)=0이 아니고 n(FT)=0, n(SD)=0이면 [단계 1]로 가라.
5) n(R)=0이 아니고 n(FT)=0이 아니고 n(SD)=0이면 [단계 7]로 가라.
6) n(R)=0이 아니고 n(FT)=0, n(SD)=0이 아니면 [단계 8]로 가라.
7) n(R)=0이 아니고, n(FT)=0이 아니고, n(SD)=0이 아니면 [단계 3]을 반복하여 수행한다.

[단계 4] 배열된 job의 평균처리시간을 구하고 끝마친다.

[단계 5] FT에 dummy job을 두고 [단계 3]으로 간다.

[단계 6] SD에 dummy job를 두고 [단계 3]으로 간다.

[단계 7] 남아 있는 job type중 FT에 있는 job과 Job Matching이 되는 job중에 CT가 가장 작은 job의 갯수가 많은 것을 먼저 택한다. [단계 3]으로 간다.

[단계 8] 남아 있는 job type중 SD에 있는 job과 Job Matching이 되는 job중에 CT가 가장 작은 job을 선택하여 FT에 두고, 만약 CT가 같은 것이 두개 이상이면 job의 갯수가 많은 것을 먼저 택한다. [단계 3]으로 간다.

3.5 CS=2일 때 SCTJMF Algorithm을 적용한 예제

본 연구에서 모델로 이용한 자동생산체제(FMS)의 구성여건이 다음과 같을때의 SCTJMF Algorithm을 적용하여 예제를 풀어 본다.

1) M=3　　2) CS=2　　3) P=10개
4) ①Jaba : 5개,　②ja : 10개,　③Jb : 5개

[단계 1] FT=②Ja

[단계 2] SD=③Jb

[단계 3] FT에 있는 Ja를 먼저 SD에 있는 Jb를 나중에 배열한다.

[그림5a] A SCTJMF Algorithm Schedule for CS=2

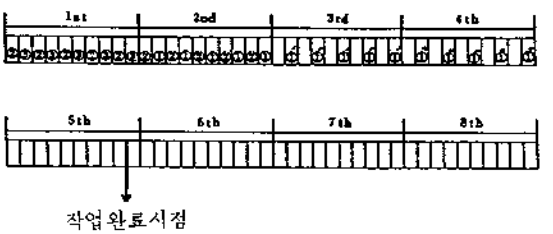
n(R)=0이 아니고, n(FT)=0이 아니고, n(SD)=0이 아니므로 반복해서 배열한다.

[그림5b] A SCTJMF Algorithm Schedule for CS=2

n(R)=0이 아니고, n(FT)=0이 아니고, n(SD)=0이므로 [단계 7]로 간다.

[단계 7] SD에 Jcba두고 [단계 3]으로 가라.

[단계 3] FT에 있는 Ja를 먼저 SD에 있는 Jcba를 나중에 배열한다.



[그림5c] A SCTJMF Algorithm Schedule for CS=2

n(R)=0, n(FT)=0, n(SD)=0이면 [단계 4]로 가라.

[단계 4] 평균처리시간을 구한다.

$F1=41+43+45+47+49=225$

$F2=10+12+14+16+18+20+22+24+26+28=190$

$F3=1+13+15+17+19=75$

$N=1+2+3=20$

$F=1+2+3=225+190+75=490$

$\bar{F}=F/(N^*CS)=490/20^*2)=12.3$

앞의 [그림3], [그림4], [그림5c]에서 보는 바와 같이 SCTF Algorithm을 적용한 Schedule과 Random Schedule을 SCTJMF Algorithm을 적용한 Schedule를 비교해 보면 아래 있는 [표3]과 같다.

Mean Flow Fime for CS=2			Evaluation by Reduction Rate	
Random Schedule ($\bar{F}r$)	SCTF Algorithm ($\bar{F}s$)	SCTJMF Algorithm ($\bar{F}*$)	$\frac{\bar{F}r-\bar{F}*}{\bar{F}r}\times100$	$\frac{\bar{F}s-\bar{F}*}{\bar{F}r}\times100$
16.7	14.7	12.3	26.3%	17.4%

〔표 3〕

An Evaluation of The Heuristic Algorithm by Reduction Rate

[표3]에서 보는 바와 같이 CS=2일 때 평균처리시간을 최소화하기 위해서 SCTJMF Algorithm이 SCTF Algorithm이나 Random Schedyle보다 훨씬 효과적임을 알 수 있다.

4. 발견적 기법의 평가

일반적으로 일정계획문제는 복잡한 조합의 문제이기 때문에 job수나 기계수가 증가하면 가능한 경우의 수가 거의 무한대에 가까워져서 열거법(Permutation Schedule)에 의해서 최적해를 구하는 것은 사실상 불가능하다.

따라서 실제 상황에서 문제에 대한 해를 쉽게 빨리 얻기 위하여 최적해에 근사한 발견적 기법의 개발에 대한 필요성이 증대되고 있으나 그 유효성을 평가하는 절대적인 방법이 아직은 개발되지 않았다[18].

본 연구에서는 하나의 대안으로 개발한 발견적기법에 의해서 얻은 해를 FORTRAN으로 프로그램하고 VAX-11/8700 컴퓨터를 이용하여 CS=1일 때의 최적해를 제공한 SCTF Algorithm 및 랜덤스케줄에 의해서 얻은 해와 비교하여 그 유효성을 평가하였다.

4.1 발생빈도에 따른 평가

본 연구에서 개발한 발견적기법의 유효성을 평가하기 위해서 Machine Center M=3, 컨베이어 속도 CS=2, 팔레트 P=100개인 상황에서 job type의 수를 1개에서 15개까지 2개씩 증가시키면서 각 경우당 7종류씩, 그리고 동시에 전체의 job수를 500개에서 1,000개까지 100개씩 증가시키면서 문제당 100번씩 112개의 문제를 시험해 본 결과 발견적 기법인 SCTJMF Algorithm에 의해 구한 작업순서에 따른 평균처리시간이 SCTJMF Algorithm보다는 95.5%, 랜덤스케줄에 의해 구한 평균 처리시간보다는 99.1%에 해당하는 111문제에서 우수한 발생빈도를 나타내었고, 1문제만이 같은 값을 나타내었다.

특히 랜덤스케줄과의 비교에서는 같은 값을 갖는 1 문제를 제외하고는 매 문제당 발견적 기법이 100%의 우수한 값을 나타내었다. 또한 발견적인 기법을 적용해서 평균 처리시간을 구할 때 job type의 수나 전체의 job수 에는 거의 영향을 받지 않으나 주어진 job type들 간에 job Matching이 이루어지는 job들끼리 그룹을 지어서 loading하면 훨씬 효과적으로 평균처리시간을 줄일 수 있다.

4.2 단축비에 의한 평가

4.1절에서 설명한 것과 동일한 방법으로 시험해 본 결과 본 연구에서 개발한 발견적 기법인 SCTJMF Algorithm이 컨베이어 속도가 1일 때 최적해를 제공하는 SCTJMF Algorithm보다는

평균 19.6%, 랜덤스케줄 보다는 1문제만이 같은 값을 나타내었고, 나머지 111개의 문제에서 평균 28.3%만큼 평균처리시간을 단축할 수 있었다.

단축비에 있어서도 발생빈도와 마찬가지로 발견적인 기법을 적용해서 평균처리시간을 구할 때 job type의 수나 전체의 job 수에는 거의 영향을 받지 않으나 job Matching이 이루어지는 job들끼리 그룹을 지어서 loading하면 훨씬 효과적으로 평균처리시간을 줄일 수 있다.

5. 결 론

본 연구에서는 제품에 대한 수요가 갈수록 다양해지고 제품의 수명기간이 점점 짧아지는 상황에서 가능한한 재공품을 줄이고, 고객의 다양한 요구와 수요변화에 능동적으로 대처해 나감과 동시에 생산성과 효율을 높일 수 있는 자동생산 체제에서 평균처리시간을 최소화하기 위하여 모델을 설정하고 분석 및 고찰하였다.

순환형 컨베이어가 장착된 자동생산 체제에서 컨베이어 속도가 1일 때 평균 처리시간을 최소화하는 최적 알고리듬과, 컨베이어 속도가 2일 때의 발견적 기법인 SCTJMF Algorithm을 개발하고 그 유효성을 평가하기 위해 Machine Center가 3개이고 팔레트 수가 100개일 때에 문제당 100번씩 112개의 문제를 시험해 본 결과 컨베이어 속도가 1일 때 최적해를 제공하는 SCTJMF Algorithm보다 95.5%, 랜덤스케줄보다는 99.1%의 우수한 발생빈도를 나타냈으며, 단축비에서는 발견적기법인 SCTJMF Algorithm이 SCTJMF Algorithm을 적용한 경우의 평균처리시간보다는 평균 19.6%, 랜덤스케줄을 적용한 경우의 평균처리시간보다 평균 28.3%를 줄일 수 있었다

앞으로 컨베이어가 정착된 자동생산 체제에서 다른 모델, 다른 평가기준에 대하여 최적해를 구하기 위한 연구와 더욱 효율성이 좋은 근사최적해를 구하기 위한 연구가 필요할 것으로 생각된다.

References

1. Baker. K.R., *Introduction to Sequencing and Scheduling,* Wiley, New York, 1974.
2. Buazacott, J.A. and Shanthikumar, J.G., "Models for understanding FMS, *AIIE Transactions,* Vol. 24, No. 4, pp. 339–350, 1980.
3. Carrie, A.S., Adhami, E., Stephens, A., and Murdoch, I.C., "*Int. J. Prod Res.,* Vol. 22, No. 6, pp. 907–916, 1985.
4. Galouzea, M. and Siegel, P., Simulation d'une ligne flexible d'assemblage, Research report, Laboratoire d'automatique, Ecole Nationale Syperieure D'Ingenieurs Electriciens de Grenoble, *Flexible Manufacturing : Methods and Studies,* Elsevier Science Publishers B.V. (North-Holland), France, 1984.
5. Hartley, John, *FMS at work,* IFS (Publications) Ltd., 1984.
6. Klahorst, H.T., "Howto Plan Your FMS," *Manufacturing Eng.,* Vol. 91, No. 3, pp. 52–54, 1985.
7. Kusiak, A., "FMS : A Structural Approach," *Int. J. Prod. Res.,* Vol. 23, Np. 6, pp. 1057–1073, 1985.
8. Kusiak, A., "Application of OR Models and Techniques in FMS.," *European J. of OR,* Vol. 24, pp. 336-345, 1986.
9. Kusiak, A., "Scheduling Flexible Maching and Assembly System," *Proc. Second ORSA/TIMS Conf. on FMS : OR Models and Applications,* Ann Arbor, MI, pp. 521–532, 1986.
10. Nof Shimon Y., Barash Moshe M., ans Solberg James J., "Operational Control of Item Flow in Versatile Manufacturing Systems," *Int, J. Prod. Res.,* Vol. 17, No. 5, pp. 479–489, 1979.
11. Ranky, P., *The Design and Operation of FMS,* IFS (Publications) Ltd., 1983.
12. Richard, R.J., "Scheduling FMS Using Mean Value Analysis," *Proc. of IEEE Conf. on Decision and Control,* pp. 701–706, 1980.
13. Robert, J. Wittrock, "Scheduling Algorithm for Flexible Flow Lines," *IBM. H. Res. Develop.,* Vol. 29, No. 4, 1985.
14. Sriskandarajah, C., Ladet. P., and Germain, R., "Scheduling Methods for Manufacturing Systems," *Flexible Manufacturing : Methods and Studies,* Elsevier Science Publishers B.V. (North-Holland), 1986.
15. Stecke, K.E. and Solberg, J., "Loading and Control Policies for a FMS," *Int. J. Prod. Res.,* Vol. 19, Np. 5, pp. 481–490, 1981.
16. Stecke, K.E., "A Hierarchcal Approach to Solving Machine Groupping and Loading problems of FMS," *European J. of OR,* Vol. 24, pp. 369–378, 1986.
17. Van Looveren. A.J., Gelders, L.F., and Van Wasenhove, L.N., "A Review of Planning odels," *Modelling and Design of FMS,* Elsevier Science Publishers B.V., 1986.
18. Zanskis, S.H. and Evans J.R., "Heuristic Optimization : Why, When and How to use it," *Interfaces,* Vol. 11, Np. 5, pp. 84–90, 1981.
19. French, S., *Sequencing and Scheduling,* Ellis Horwood Limited, New York, 198 .